Dmytro Horyslavets
Kyiv Academic University

Introduction

The amount of biological sequencing data available in public repositories is growing exponentially. Even for defined subsets of samples that are collected within large-scale studies, the entire sequencing data of a study can comprise hundreds of terabytes. Using vast raw sequencing data makes its analysis complicated and decreases its accessibility to the broader research community. Thus, for efficient and scalable analysis, a more sophisticated approach is required. The MetaGraph [1] framework transforms large repositories of sequencing data into compressed and accessible representations (indexes), which can be efficiently queried with any sequence of interest. The MetaGraph index consists of the de Bruijn graph and its annotation metadata, allowing for sequence search and assembly.

Project Proposal

The project goal is to extend the MetaGraph framework by adding an option to extract reads from the Counting de Bruijn Graphs, which appear during the index construction in the MetaGraph workflow. That is, given a sequence search result, extract all reads that overlap the parts of the graph that matched.

That will help to perform more refined graph cleaning during the assembly step and also be useful for further downstream analysis after the sequence search.

Software deliverables

The MetaGraph is created mostly using the C++ language. Its source code is stored in the GitHub repository. The new features planned to be included in the framework during this project will also be implemented using the C++ language.

Timeline

The project work will be arranged for 10 weeks starting from July 5th to September 12th under the supervision of Andre Kahles (University of Zurich).

Week 1-2

Introduction to the MetaGraph framework source code. Getting familiar with its concepts and implementation aspects. Literature review on Counting de Bruijn graphs.

Week 3-5

Extending the MetaGraph annotator to mark sequence end-points during index construction. Implementing an algorithm to extract reads from the assembled de Bruijn graph. Evaluating the algorithm efficiency.

Week 6-8

Testing the implemented feature of the read extraction. Inserting it in the general MetaGraph pipeline as an option in the graph cleaning step.

Week 9-10

Final testing and code refactoring. Finalizing the project and creating a summary presentation.

References

[1] Karasikov, M., Mustafa, H., Danciu, D., Zimmermann, M., Barber, C., Rätsch, G., & Kahles, A. (2020). Metagraph: Indexing and analysing nucleotide archives at petabase-scale. BioRxiv.